

Approximation Algorithm for Minimum Weight Connected m -Fold Dominating Set

Zhao Zhang¹ Jiao Zhou² Ker-I Ko³ Ding-zhu Du⁴

¹ *College of Mathematics Physics and Information Engineering, Zhejiang Normal University
Jinhua, Zhejiang, 321004, China*

² *College of Mathematics and System Sciences, Xinjiang University
Urumqi, Xinjiang, 830046, China*

³ *Department of Computer Science, National Chiao Tung University
Hsinchu, 30050, Taiwan*

⁴ *Department of Computer Science, University of Texas at Dallas
Richardson, Texas, 75080, USA*

Abstract: Using connected dominating set (CDS) to serve as a virtual backbone in a wireless networks can save energy and reduce interference. Since nodes may fail due to accidental damage or energy depletion, it is desirable that the virtual backbone has some fault-tolerance. A k -connected m -fold dominating set $((k, m)$ -CDS) of a graph G is a node set D such that every node in $V \setminus D$ has at least m neighbors in D and the subgraph of G induced by D is k -connected. Using (k, m) -CDS can tolerate the failure of $\min\{k - 1, m - 1\}$ nodes. In this paper, we study Minimum Weight $(1, m)$ -CDS problem $((1, m)$ -MWCDs), and present an $(H(\delta + m) + 2H(\delta - 1))$ -approximation algorithm, where δ is the maximum degree of the graph and $H(\cdot)$ is the Harmonic number. Notice that there is a $1.35 \ln n$ -approximation algorithm for the $(1, 1)$ -MWCDs problem, where n is the number of nodes in the graph. Though our constant in $O(\ln \cdot)$ is larger than 1.35, n is replaced by δ . Such a replacement enables us to obtain a $(6.67 + \varepsilon)$ -approximation for the $(1, m)$ -MWCDs problem on unit disk graphs.

Keyword: m -fold dominating set, connected dominating set, non-submodular function, greedy algorithm, unit disk graph.

1 Introduction

A wireless sensor network (WSN) consists of spatially distributed autonomous sensors to monitor physical or environmental condition, and to cooperatively pass their data through the network. During recent years, WSN has been widely used in many fields,

such as environment and habitat monitoring, disaster recovery, health applications, etc. Since there is no fixed or predefined infrastructure in WSNs, frequent flooding of control messages from sensors may cause a lot of redundant contentions and collisions. Therefore, people have proposed the concept of virtual backbone which corresponds to a connected dominating set in a graph (Das and Bhargharan [4] and Ephremides *et al.* [8]).

Given a graph G with node set V and edge set E , a subset of nodes $C \subseteq V$ is said to be a *dominating set* (DS) of G if any $v \in V \setminus C$ is adjacent to at least one node of C . We say that a dominating set C of G is a *connected dominating set* of G if the subgraph of G induced by C , denoted by $G[C]$, is connected. Nodes in C are called *dominators*, the nodes in $V \setminus C$ are called *dominatees*.

Because sensors in a WSN are prone to failures due to accidental damage or battery depletion, it is important to maintain a certain degree of redundancy such that the virtual backbone is more fault-tolerant. In a more general setting, every sensor has a cost, it is desirable that under the condition that tasks can be successfully accomplished, the whole cost of virtual backbone is as small as possible. These considerations lead to the Minimum Node-Weighted k -Connected m -Fold Dominating Set problem (abbreviated as (k, m) -MWCDS), which is defined as follows:

Definition 1.1 ((k, m) -MWCDS). Let G be a connected graph, k and m be two positive integers, $c : V \rightarrow R^+$ be a cost function on nodes. A node subset $D \subseteq V$ is an m -fold dominating set (m -DS) if every node in $V \setminus D$ has at least m neighbors in D . It is a k -connected m -fold dominating set ((k, m) -CDS) if furthermore, the subgraph of G induced by D is k -connected. The (k, m) -MWCDS problem is to find a (k, m) -CDS D such that the cost of D is minimized, that is, $c(D) = \sum_{u \in D} c(u)$ is as small as possible.

After Dai and Wu [3] proposed using (k, k) -CDS as a model for fault-tolerant virtual backbone, a lot of approximation algorithms emerged, most of which are on unit disk graphs. In a unit disk graph (UDG), every node corresponds to a sensor on the plane, two nodes are adjacent if and only if the Euclidean distance between the corresponding sensors is at most one unit. There are a lot of studies on fault-tolerant virtual backbone in UDG which assume unit weight on each disk. However, for a general graph with a general weight function, related studies are rare.

In this paper, we present a $(H(\delta + m) + 2H(\delta - 1))$ -approximation algorithm for the $(1, m)$ -MWCDS problem on a general graph, where δ is the maximum degree of the graph, and $H(\gamma) = \sum_{i=1}^{\gamma} 1/i$ is the Harmonic number. It is a two-phase greedy algorithm. First, it constructs an m -fold dominating set D_1 of G . Then it connects D_1 by adding a set of connectors D_2 . It is well known that if the potential function and the cost function related with a greedy algorithm is monotone increasing and submodular, then an $O(\ln n)$ performance ratio can be achieved. Unfortunately, for various minimum CDS problems, no such potential functions are known. Nevertheless, we manage, in this paper, to deal with a nonsubmodular potential function and achieve an $O(\ln \delta)$ performance ratio.

It should be pointed out that for a general graph, Guha and Khuller [10] proposed a $1.35 \ln n$ -approximation for the $(1, 1)$ -MWCDS problem. Though our constant in $O(\ln n)$

is larger, the parameter n in the performance ratio is replaced by δ . In many cases, δ might be substantially smaller than n . In particular, for an UDG, due to such a replacement, after having found an m -DS, the connecting part has a performance ratio at most 3.67. Combining this with the $(3 + \varepsilon)$ -approximation algorithm for the Minimum Weight m -DS problem in UDG [26], we can obtain a performance ratio at most $6.67 + \varepsilon$ for the $(1, m)$ -MWCDS problem in UDG. In [30], Zou *et al.* proposed a method for the connecting part with performance ratio at most 3.875, which makes use of a 1.55-approximation algorithm [14] for the classic Minimum Steiner Tree problem. If using currently best ratio for Minimum Steiner Tree problem [1], then their ratio is at most 3.475. Notice that the algorithm in [1] uses randomized iterative rounding. So, although our ratio 3.67 is a little larger than 3.475, our algorithm has the advantage that it is purely combinatorial. Furthermore, we believe that our method is also of more theoretical interests and may find more applications in the study of other related problems.

The rest of this paper is organized as follows. Section 2 introduces related works. Some notation and some preliminary are given in Section 3. In Section 4, the algorithm is presented. Section 5 analyzes the performance ratio. Section 6 improves the ratio on unit disk graph. Section 7 concludes the paper.

2 Related work

The idea of using a connected dominating set as a virtual backbone for WSN was proposed by Das and Bhargharan [4] and Ephremides *et al.* [8]. Constructing a CDS of the minimum size is NP-hard. In fact, Guha and Khuller [10] proved that a minimum CDS cannot be approximated within $\rho \ln n$ for any $0 < \rho < 1$ unless $NP \subseteq DTIME(n^{O(\log \log n)})$. In the same paper, they proposed two greedy algorithms with performance ratios of $2(H(\delta) + 1)$ and $H(\delta) + 2$, respectively, where δ is the maximum degree of the graph and $H(\cdot)$ is the harmonic number. This was improved by Ruan *et al.* [15] to $2 + \ln \delta$. Du *et al.* [5] presented a $(1 + \varepsilon) \ln(\delta - 1)$ -approximation algorithm, where ε is an arbitrary positive real number.

For unit disk graphs, a polynomial time approximation scheme (PTAS) was given by Cheng *et al.* [2], which was generalized to higher dimensional space by Zhang *et al.* [25]. There are a lot of studies on distributed algorithms for this problem. For a comprehensive study on CDS in UDG, the readers may refer to the book [7].

Considering the weighted version of the CDS problem, Guha and Khuller [10] proposed a $(c_n + 1) \ln n$ -approximation algorithm in a general graph, where $c_n \ln k$ is the performance ratio for the node weighted Steiner tree problem (k is the number of terminal nodes to be connected). Later, they [11] improved it to an algorithm of performance ratio at most $(1.35 + \varepsilon) \ln n$. For the minimum weight CDS in UDG, Zou *et al.* [30] gave a $(9.875 + \varepsilon)$ -approximation.

The problem of constructing fault-tolerant virtual backbones was introduced by Dai and Wu [3]. They proposed three heuristic algorithms for the minimum (k, k) -CDS prob-

lem. However, no performance ratio analysis was given. A lot of works have been done for the CDS problem in UDG. The first constant approximation algorithm in this aspect was given by Wang *et al.* [21], who obtained a 72-approximation for the $(2, 1)$ -CDS problem in UDG. Shang *et al.* [16] gave an algorithm for the minimum $(1, m)$ -CDS problem and an algorithm for the minimum $(2, m)$ -CDS problem in UDG, the performance ratios are $5 + \frac{5}{m}$ for $m \leq 5$ and 7 for $m > 5$, and $5 + \frac{25}{m}$ for $2 \leq m \leq 5$ and 11 for $m > 5$, respectively. Constant approximation algorithms also exist for $(3, m)$ -CDS in UDG [22, 23]. For general k and m , Li *et al.* proposed a γ_m -approximation for the (k, m) -CDS problem, where $\gamma_m = O((\delta(\delta - 1)^2)^{k-1})$. Notice that this is not a constant approximation.

For the fault-tolerant CDS problem in a general graph, Zhang *et al.* [27] gave a $2rH(\delta_r + m - 1)$ -approximation for the minimum r -hop $(1, m)$ -CDS problem, where δ_r is the maximum degree in G^r , the r -th power graph of G . A node u is r -hop dominated by a set D if it is at most r -hops away from D . In particular, taking $r = 1$, the algorithm in [27] has performance ratio at most $2H(\delta + m - 1)$ for the minimum $(1, m)$ -CDS problem. This was improved by our recent work [29] to $2 + H(\delta + m - 2)$. We [17] also gave a $(4 + \ln(\delta + m - 2) + 2 \ln(2 + \ln(\delta + m - 2)))$ -approximation algorithm for the minimum $(2, m)$ -CDS problem on a general graph.

For the weighted version of fault-tolerant CDS problem on UDG, we [26] recently proposed a $(3 + \varepsilon)$ -approximation for the minimum weight m -fold dominating set problem. Combining the connecting part of paper [30], $(1, m)$ -MWCDS on UDG has a $(6.875 + \varepsilon)$ -approximation. We [28] also studied the $(2, m)$ -MWCDS problem on UDG, achieving an performance ratio of $(8 + \varepsilon)$.

As far as we know, there is no previous work on the approximation of the weighted version of fault-tolerant CDS problem in a general graph.

Notice that in [30], the 3.875-approximation for the connecting part is based on the 1.55-approximation algorithm [14] for the classic minimum Steiner tree problem. If the best known ratio for the Steiner tree problem is used, which is 1.39 currently, then their connecting part has performance ratio at most 3.475. Notice that the 1.39-approximation for the Steiner tree problem uses randomized iterative rounding. So, although our performance ratio is larger than 3.475, it has the advantage that it is purely combinatorial.

3 Preliminaries

In this section, we introduce some notation and give some preliminary results. For a node $u \in V(G)$, denote by $N_G(u)$ the set of neighbors of u in G , and $\deg_G(u) = |N_G(u)|$ is the degree of node u in G . For a node subset $D \subseteq V(G)$, $N_G(D) = \bigcup_{u \in D} N_G(u) \setminus D$ is the neighbor set of D , $G[D]$ is the subgraph of G induced by D . When there is no confusion in the context, the vertex set of a subgraph will be used to denote the subgraph itself.

For an element set U , suppose $f : 2^U \mapsto \mathbb{R}^+$ is a set function on U (called a *potential*

function). For two element sets $C, D \subseteq U$, let

$$\Delta_D f(C) = f(C \cup D) - f(C)$$

be the marginal profit obtained by adding D into C . For simplicity, $\Delta_u f(C)$ will be used to denote $\Delta_{\{u\}} f(C)$ when u is a node. Potential function f is *monotone increasing* if $f(C) \leq f(D)$ holds for any subsets $C \subseteq D \subseteq V$. It is *submodular* if and only if $\Delta_u f(C) \geq \Delta_u f(D)$ holds for any $C \subseteq D \subseteq V$ and any $u \in V \setminus D$. A monotone increasing and submodular function f with $f(\emptyset) = 0$ is called a *polymatroid*. Given an element set U with cost function $c : U \mapsto \mathbb{R}^+$ and given a polymatroid $f : 2^U \mapsto \mathbb{R}^+$, denote by $\Omega_f = \{C \subseteq U : \Delta_u f(C) = 0 \text{ for any } u \in U\}$. The *Submodular Cover* problem is:

$$\begin{aligned} \min \quad & c(C) = \sum_{u \in C} c(u) \\ \text{s.t.} \quad & C \in \Omega_f. \end{aligned}$$

The following is a classic result which can be found in [6] Theorem 2.29.

Theorem 3.1. *The greedy algorithm for the submodular cover problem has performance ratio $H(\gamma)$, where $\gamma = \max\{f(\{u\}) : u \in U\}$ and $H(\gamma) = \sum_{i=1}^{\gamma} 1/i$ is the Harmonic number.*

4 The Algorithm

4.1 The Algorithm

For a node set C , denote by $p(C)$ the number of components in $G[C]$. For a node $u \in V \setminus C$, let $N_C(u)$ denote the set of neighbors of u in C . Define

$$q_C(u) = \begin{cases} \max\{m - |N_C(u)|, 0\}, & u \in V \setminus C, \\ 0, & u \in C, \end{cases}$$

and

$$q(C) = m|V| - \sum_{u \in V} q_C(u).$$

For a node set $U \subseteq V$, denote by $NC_C(U)$ the set of components of $G[C]$ which are adjacent to U . Every component in $NC_C(U)$ is called a *component neighbor* of U in $G[C]$. For a node $u \in V$, we shall use S_u to denote some star with center u , that is, S_u is a subgraph of G induced by edges between node u and some of u 's neighbors in G . In particular, a node is a star of cardinality one and an edge is a star of cardinality two. To abuse the notation a little, we also use S_u to denote the set of nodes in S_u . Suppose $S_u \setminus \{u\} = \{u_1, u_2, \dots, u_s\}$, where $c(u_1) \leq c(u_2) \leq \dots \leq c(u_s)$. Define

$$p'_C(S_u) = |NC_C(u)| - 1 + \sum_{i=1}^s \min\{1, -\Delta_{u_i} p(C \cup \{u, u_1, \dots, u_{i-1}\})\}. \quad (1)$$

Call $e_C(S_u) = p'_C(S_u)/c(S_u)$ the *efficiency of S_u with respect to C* .

The algorithm is presented in Algorithm 1.

Algorithm 1

Input: A connected graph $G = (V, E)$.

Output: A $(1, m)$ -CDS D_G of G .

```

1: Set  $D_1 \leftarrow \emptyset$ 
2: while there exists a node  $u \in V \setminus D_1$  such that  $\Delta_u q(D_1) > 0$ , do
3:   select  $u$  which maximizes  $\Delta_u q(D_1)/c(u)$ 
4:    $D_1 \leftarrow D_1 \cup \{u\}$ 
5: end while
6: Set  $D_2 \leftarrow \emptyset$ 
7: while there exists a star  $S_u \subseteq V \setminus (D_1 \cup D_2)$  such that  $p'_{D_1 \cup D_2}(S_u) > 0$ , do
8:   select a star  $S_u$  with the largest efficiency with respect to  $D_1 \cup D_2$ .
9:    $D_2 \leftarrow D_2 \cup S_u$ 
10: end while
11: Output  $D_G \leftarrow D_1 \cup D_2$ 

```

4.2 The Idea of the Algorithm

The idea underlying the algorithm is as follows. Potential function $q_{D_1}(u)$ measures how many more times that node u needs to be dominated, and $q(D_1)$ is the total residual domination requirement. As can be seen from Lemma 5.2, at the end of the first phase, we have an m -fold dominating set D_1 . Then, the second phase aims to connect it by adding a connector set D_2 . A natural potential function for connection is $p(D_1 \cup D_2)$, the number of components of $G[D_1 \cup D_2]$. That is, every iteration chooses a node set S to be added into D_2 which reduces the number of components by the largest amount until $p(D_1 \cup D_2)$ reaches 1. It is a folklore result (see Lemma 5.5) that simultaneously adding at most two nodes can reduce the number of components (even when adding any single node does not reduce the number of components). So, it is natural to use

$$\max\{-\Delta_S p(D_1 \cup D_2)/c(S) : S \subseteq V \setminus (D_1 \cup D_2), 1 \leq |S| \leq 2\} \quad (2)$$

to work as a criterion for the choice of node set S to be added into D_2 . However, choosing at most two nodes might yield a solution with very bad performance ratio. Consider the example shown in Fig.1, its optimal solution $OPT = \{u, v_1, \dots, v_d\}$ has cost $opt = 1 + (d+1)\varepsilon$. If (2) is used as the greedy criterion, then the output is $\bigcup_{i=1}^d \{u_i, v_i\}$, whose cost is $c(\bigcup_{i=1}^d \{u_i, v_i\}) = d(1 + \varepsilon) \approx d \cdot opt = \frac{n-2}{3}opt$. To overcome such a shortcoming, an idea is to choose some star S_u to maximize $-\Delta_{S_u} p(D_1 \cup D_2)/c(S_u)$ in each iteration. However, the computation of a most efficient star will take exponential time. This is why we define p' as in (1) to be used in the greedy criterion. On one hand, p' also plays the role of counting the reduction on components. On the other hand, a most efficient S_u to maximize $p'_{D_1 \cup D_2}(S_u)/c(S_u)$ can be found in polynomial time, which will be shown in Subsection 5.2.

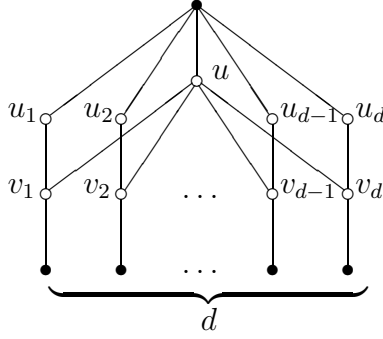


Figure 1: Solid nodes represent nodes in C . The costs on circled nodes are $c(u_1) = c(u_2) = \dots = c(u_d) = 1$, $c(v_1) = c(v_2) = \dots = c(v_d) = \varepsilon$, and $c(u) = 1 + \varepsilon$.

5 The Analysis of the Algorithm

In this section, we analyze the performance ratio of Algorithm 1.

5.1 The Analysis of D_1

Lemma 5.1. *Function q is a polymatroid.*

Proof. Obviously, $q(\emptyset) = 0$. It is easy to see that for any node $u \in V$, function $-q_C(u)$ is monotone increasing and submodular with respect to C . So, function q , being the summation of a constant function and some monotone increasing and submodular functions, is also monotone increasing and submodular. \square

Lemma 5.2. *The final node set D_1 in Algorithm 1 is an m -fold dominating set of G .*

Proof. If there exists a node $u \in V \setminus D_1$ with $|N_{D_1}(u)| < m$, then $-\Delta_u q_{D_1}(u) = -q_{D_1 \cup \{u\}}(u) + q_{D_1}(u) = -0 + (m - |N_{D_1}(u)|) > 0$, and $-\Delta_u q_{D_1}(v) \geq 0$ for any node $v \in V \setminus \{u\}$ (by the monotonicity of $-q_{D_1}(v)$ with respect to D_1). Thus $\Delta_u q(D_1) = -\sum_{v \in V} \Delta_u q_{D_1}(v) > 0$, and the algorithm does not terminate at this stage. \square

Theorem 5.3. *The final node set D_1 in Algorithm 1 has weight $w(D_1) \leq H(m + \delta) \cdot \text{opt}$, where δ is the maximum degree of graph G , and opt is the optimal value for the m -MWCDs problem.*

Proof. By Lemma 5.1 and Lemma 5.2, the minimum weight m -fold dominating set problem is a special submodular cover problem with potential function q . So by Theorem 3.1, we have $w(D_1) \leq H(\gamma) \cdot \text{opt}'$, where opt' is the optimal value for the m -MWDS problem. Then, the result follows from the observation that $\text{opt}' \leq \text{opt}$ and $\gamma = \max_{u \in V} \{q(\{u\})\} = \max_{u \in V} \{m|V| - \sum_{v \in V \setminus \{u\}} (m - |N_{\{u\}}(v)|)\} = \max_{u \in V} \{m + d_G(u)\} = m + \delta$. \square

5.2 The Computation of an Optimal Star for Greedy Choice

The idea for the definition of $p'_C(S_u)$ is as follows: Adding node u into C will merge those components in $NC_C(u)$ into one component of $G[C \cup \{u\}]$, say \tilde{C} , which reduces the number of components by $|NC_C(u)| - 1$. Then adding nodes in $S_u \setminus \{u\}$ sequentially according to the increasing order of their costs. Notice that

$$-\Delta_{u_i} p(C \cup \{u_1, \dots, u_{i-1}\}) = |NC_C(u_i) \setminus NC_C(u, u_1, \dots, u_{i-1})| \quad (3)$$

is the number of components newly merged into \tilde{C} . So, the term in the summation of definition (1) indicates that if adding u_i merges at least one more component, we regard its contribution to $p'_C(S_u)$ as one. The advantage of such a counting is that an optimal star for the greedy criterion is polynomial-time computable. This claim is based on the following lemma. A simple relation will be used in the proof: for four positive real numbers a, b, c, d

$$\frac{a+b}{c+d} \geq \frac{b}{d} \Rightarrow \frac{a}{c} \geq \frac{b}{d}. \quad (4)$$

Lemma 5.4. *Suppose C is a dominating set of graph G . Then, there exists a most efficient star S_u with respect to C such that $|NC_C(v)| = 1$ for every node $v \in S_u \setminus \{u\}$. Furthermore, if we denote by C_v the unique component in $NC_C(v)$, then components in $\{C_v\}_{v \in S_u \setminus \{u\}}$ are all distinct and they are also distinct from those components in $NC_C(u)$.*

Proof. Suppose S_u is a most efficient star with $S_u \setminus \{u\} = \{u_1, \dots, u_s\}$ such that $c(u_1) \leq \dots \leq c(u_s)$. We first show that for any $i = 1, \dots, s$,

$$-\Delta_{u_i} p(C \cup \{u, u_1, \dots, u_{i-1}\}) \geq 1. \quad (5)$$

Suppose this is not true. Let i be the first index with $-\Delta_{u_i} p(C \cup \{u, u_1, \dots, u_{i-1}\}) = 0$. Then by (3), we have $NC_C(u_i) \subseteq NC_C(u, u_1, \dots, u_{i-1})$ and thus $NC_C(u, u_1, \dots, u_{i-1}) = NC_C(u, u_1, \dots, u_i)$. It follows that for any $j > i$,

$$-\Delta_{u_j} p(C \cup \{u, u_1, \dots, u_{i-1}, u_{i+1}, \dots, u_{j-1}\}) = -\Delta_{u_i} p(C \cup \{u, u_1, \dots, u_{j-1}\}).$$

So, for the star $S'_u = S_u \setminus \{u_i\}$, we have $p'_C(S'_u) = p'_C(S_u)$ and thus $p'_C(S'_u)/c(S'_u) > p'_C(S_u)/c(S_u)$, contradicting the maximality of $p'_C(S_u)/c(S_u)$.

As a consequence of (5),

$$p'_C(S_u) = |NC_C(u)| - 1 + s.$$

Suppose there is a node u_i with $|NC_C(u_i)| \geq 2$, we choose u_i to be such that i is as small as possible. Let $S'_u = S_u \setminus \{u_i\}$. Notice that for any $j > i$,

$$-\Delta_{u_j} p(C \cup \{u, u_1, \dots, u_{i-1}, u_{i+1}, \dots, u_{j-1}\}) \geq -\Delta_{u_i} p(C \cup \{u, u_1, \dots, u_{j-1}\}).$$

Combining this with (5), we have

$$p'_C(S'_u) = |NC_C(u)| - 1 + (s - 1) = p'_C(S_u) - 1.$$

By the maximality of S_u , we have

$$\frac{p'_C(S'_u)}{c(S'_u)} \leq \frac{p'_C(S_u)}{c(S_u)} = \frac{p'_C(S'_u) + 1}{c(S'_u) + c(u_i)}.$$

Then by (4),

$$\frac{p'_C(S'_u)}{c(S'_u)} \leq \frac{1}{c(u_i)}.$$

It follows that

$$\frac{p'_C(S_u)}{c(S_u)} = \frac{p'_C(S'_u) + 1}{c(S'_u) + c(u_i)} \leq \frac{\frac{1}{c(u_i)}c(S'_u) + 1}{c(S'_u) + c(u_i)} = \frac{1}{c(u_i)} \leq \frac{p'_C(u_i)}{c(u_i)},$$

where the last inequality holds because $p'_C(u_i) = |NC_C(u_i)| - 1 \geq 1$. Hence u_i is also a most efficient star. It is a trivial star satisfying the requirement of the lemma.

Next, suppose S_u is a nontrivial star in which every $v \in S_u \setminus \{u\}$ has $|NC_C(v)| = 1$. Notice that an equivalent statement of (5) is that $|NC_C(u_i) \setminus NC_C(u, u_1, \dots, u_{i-1})| \geq 1$ for any $i = 1, \dots, s$. The second part of this lemma follows. \square

By Lemma 5.4, a most efficient star with respect to C can be found in the following way. Guessing the center of the star requires time $O(n)$. Suppose u is the guessed center. Let $N^{(u)} = \{v : v \text{ is a neighbor of } u \text{ in } G \text{ and } |NC_C(v)| = 1\}$. Order the nodes in $N^{(u)}$ as u_1, \dots, u_s such that $c(u_1) \leq \dots \leq c(u_s)$. For $i = 1, \dots, s$, scan u_i sequentially. If u_i has $NC_C(u_i) \subseteq NC_C(u, u_1, \dots, u_{i-1})$, then remove it from $N^{(u)}$. For convenience of statement, suppose the remaining set $N^{(u)} = \{u_1, u_2, \dots, u_t\}$. Then, the node set of a most efficient star centered at u must be of the form $\{u, u_1, u_2, \dots, u_l\}$ for some $l \in \{0, \dots, t\}$. So, it suffices to compute the efficiency of the $t + 1$ sets $\{u, u_1, \dots, u_l\}$ for $l = 0, \dots, t$ and choose the most efficient one from them. Clearly, such a computation can be done in polynomial time.

5.3 Correctness of the Algorithm

The following result is a folklore in the study of CDS (see, for example, [19]).

Lemma 5.5. *Suppose D is a dominating set of G such that $G[D]$ is not connected. Then, two nearest components of $G[D]$ are at most three hops away.*

Theorem 5.6. *The output D_G of Algorithm 1 is a $(1, m)$ -CDS.*

Proof. By Lemma 5.2, D_1 is an m -fold DS, and thus D_G is also an m -DS. If $G[D_G]$ is not connected, consider two nearest components of $G[D_G]$, say G_1 and G_2 . Let $P = u_0 u_1 \dots u_t$ be a shortest path between G_1 and G_2 , where $u_0 \in V(G_1)$ and $u_t \in V(G_2)$. By Lemma 5.5, we have $t = 2$ or 3 . Then, u_1 (in the case $t = 2$) or $u_1 u_2$ (in the case $t = 3$) is a star S_{u_1} with $p'_{D_G}(S_{u_1}) > 0$. The algorithm will not terminate. \square

5.4 Decomposition of Optimal Solution

The following lemma, as well as its proof, can be illustrated by Fig.2.

Lemma 5.7. *Let G be a connected graph, C be a dominating set of G and C^* be a connected dominating set of G . Then $C^* \setminus C$ can be decomposed into the union of node sets $C^* \setminus C = Y_0 \cup Y_1 \cup Y_2 \cup \dots \cup Y_h$ such that:*

- (i) *for $1 \leq i \leq h$, subgraph $G[Y_i]$ contains a star;*
- (ii) *subgraph $G[C \cup Y_1 \cup Y_2 \cup \dots \cup Y_h]$ is connected;*
- (iii) *for $1 \leq i \leq h$, $|NC_C(Y_i)| \geq 2$;*
- (iv) *any node of $C^* \setminus C$ belongs to at most two sets of $\{Y_0, Y_1, Y_2, \dots, Y_h\}$.*

Proof. Let H be the graph obtained from $G[C \cup C^*]$ by contracting every component of $G[C]$ into a super-node (call it a terminal node). Since $G[C \cup C^*]$ is connected, H is also connected, and thus H contains a spanning tree T . Recursively pruning non-terminal leaves, we obtain a tree T' in which every leaf is a terminal node (see Fig.2(a)(b)). Let Y_0 be the set of pruned nodes. We may assume that

$$\text{every non-terminal node has at least one terminal neighbor in } T'. \quad (6)$$

In fact, if this is not true, then we can modify T' into another tree satisfying this assumption. See Fig.2(c)(d) for an illustration. In (c), node u does not have a terminal neighbor in T' . Since C is a dominating set, u is adjacent with some component of $G[C]$, say the component corresponding to terminal node v . Adding edge uv creates a unique cycle in $T' + uv$. Removing the edge on this cycle which is incident with u in T' , we have another tree in which u has a terminal neighbor (see Fig.2(d)). Notice that the removed edge is between two non-terminal nodes. So, the number of non-terminal nodes which have no terminal neighbors is strictly reduced. Recursively making such a modification results in a tree satisfying assumption (6).

Tree T' can be viewed as a Steiner tree. By splitting off at non-leaf terminal nodes, T' can be decomposed into full components T'_1, \dots, T'_l (a *full component* in a Steiner tree is a subtree in which a node is a leaf if and only if it is a terminal node, see Fig.6(e)). Let T_i be the subtree of T'_i induced by those non-terminal nodes.

For each $i \in \{1, \dots, l\}$, let v_i be an arbitrary node of T_i and view T_i as a tree rooted at v_i . For each node $v \in V(T_i)$, let S_v be the star centered at node v which contains all children of v in T_i . Let $\mathcal{S}_i = \{S_v : v \in V(T_i) \text{ and } |NC_C(S_v)| \geq 2\}$. Let $\mathcal{S} = \bigcup_{i=1}^l \mathcal{S}_i$. Then $\{Y_v = V(S_v) : S_v \in \mathcal{S}\}$ is a desired decomposition of $C^* \setminus (C \cup Y_0)$ (see Fig.6(f)).

It should be noted that property (6) is used to guarantee that no node is missed in the decomposition. For example, in Fig.6(c) which does not satisfy this assumption, if we choose x to be the root of the lower subtree, then $S_u = uy$ and $S_y = y$ are stars which does not have at least two terminal neighbors in T' , and thus they are excluded from \mathcal{S} . But then, node y does not belong to any star in the decomposition. \square

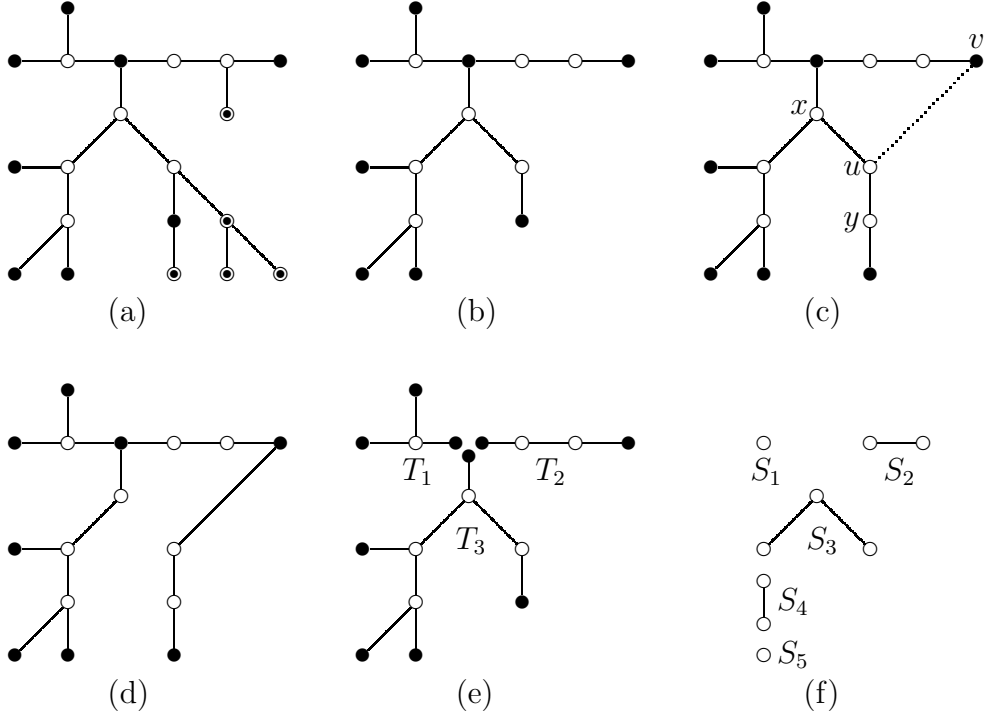


Figure 2: An example for the decomposition of $C^* \setminus C$. Solid nodes are terminal nodes which correspond to components of $G[C]$. In (a), double circle nodes are recursively pruned, yielding tree T' in (b). Figures in (c) and (d) are used to show how to obtain a tree satisfying assumption (6). In (c), the dashed edge uv is in G but not in T' . Adding edge uv and removing edge ux results in the tree in (d). Full components of T' are shown in (e). Figure (f) shows the decomposed stars.

5.5 The Performance Ratio

Theorem 5.8. *The connector set D_2 of Algorithm 1 has cost $c(D_2) \leq 2H(\delta - 1)opt$.*

Proof. Let S_1, S_2, \dots, S_g be the sets chosen by Algorithm 1 in the order of their selection into set D_2 . Let $D_G^{(0)} = D_1$. For $1 \leq i \leq g$, let

$$D_G^{(i)} = D_G^{(0)} \cup S_1 \cup \dots \cup S_i.$$

For $i = 1, \dots, g$, denote

$$r_i = -\Delta_{S_i} p(D_G^{(i-1)}) \text{ and } w_i = \frac{c(S_i)}{-\Delta_{S_i} p(D_G^{(i-1)})}.$$

Suppose $\{Y_0, Y_1, Y_2, \dots, Y_h\}$ is the decomposition of $OPT \setminus D_1$ as in Lemma 5.7, where Y_i is a star centered at node v_i . For $i = 0, 1, \dots, g$ and $j = 1, \dots, h$, denote

$$a_{i,j} = p'_{D_G^{(i-1)}}(Y_j). \quad (7)$$

For $1 \leq j \leq h$, define

$$f(Y_j) = \sum_{i=1}^g (a_{i-1,j} - a_{i,j}) w_i,$$

and let

$$f(OPT) = \sum_{j=1}^h f(Y_j).$$

Claim 1. For any $j = 1, \dots, h$, $f(Y_j) \leq H(a_{0,j} - 1)c(Y_j)$.

Since S_i is chosen according to Lemma 5.4, the special structure of S_i implies that

$$p'_{D_G^{(i-1)}}(S_i) = -\Delta_{S_i} p(D_G^{(i-1)}).$$

Hence,

$$w_i = \frac{c(S_i)}{p'_{D_G^{(i-1)}}(S_i)}.$$

Then, by the greedy choice of S_i , we have

$$w_i = \frac{c(S_i)}{p'_{D_G^{(i-1)}}(S_i)} \leq \frac{c(Y_j)}{p'_{D_G^{(i-1)}}(Y_j)} = \frac{c(Y_j)}{a_{i,j}}. \quad (8)$$

By the definition of p' , it can be seen that $a_{i,j}$ is a decreasing function on variable i . Hence $a_{i-1,j} - a_{i,j} \geq 0$. Then,

$$\begin{aligned}
f(Y_j) &\leq \sum_{i=1}^g (a_{i-1,j} - a_{i,j}) \frac{c(Y_j)}{a_{i-1,j} - 1} \\
&= c(Y_j) \sum_{i=1}^g \frac{(a_{i-1,j} - 1) - (a_{i,j} - 1)}{a_{i-1,j} - 1} \\
&\leq c(Y_j) \sum_{i=1}^g (H(a_{i-1,j} - 1) - H(a_{i,j} - 1)) \\
&= c(Y_j) (H(a_{0,j} - 1) - H(a_{g,j} - 1)),
\end{aligned}$$

where the second inequality uses the fact that for any integers $a \geq b$,

$$\frac{a-b}{a} = \sum_{l=b+1}^a \frac{1}{a} \leq \sum_{l=b+1}^a \frac{1}{l} = H(a) - H(b).$$

Observe that $a_{g,j} = 1$ since $D_G^{(g)}$ is connected, the claim follows.

Claim 2. $c(D_2) \leq f(OPT)$.

Notice that $c(D_2)$ and $f(Y_j)$ can be rewritten as

$$c(D_2) = \sum_{i=1}^g r_i w_i = \sum_{i=1}^g \left(\sum_{l=i}^g r_l - \sum_{l=i+1}^g r_l \right) w_i = \left(\sum_{i=1}^g r_i \right) w_1 + \sum_{i=2}^g \left(\sum_{l=i}^g r_l \right) (w_i - w_{i-1})$$

and

$$f(Y_j) = a_{0,j} w_1 + \sum_{i=2}^g a_{i-1,j} (w_i - w_{i-1}).$$

By the monotonicity of p' , we have $p'_{D_G^{(i-1)}}(S_i) \leq p'_{D_G^{(i-2)}}(S_i)$. Combining this with the greedy choice of S_{i-1} , we have

$$w_i = \frac{c(S_i)}{p'_{D_G^{(i-1)}}(S_i)} \geq \frac{c(S_i)}{p'_{D_G^{(i-2)}}(S_i)} \geq \frac{c(S_{i-1})}{p'_{D_G^{(i-2)}}(S_{i-1})} = w_{i-1}.$$

In other words, $w_i - w_{i-1} \geq 0$ for $i = 1, \dots, g$. So, to prove Claim 2, it suffices to prove that for $i = 1, \dots, g$,

$$\sum_{j=1}^h a_{i-1,j} \geq \sum_{l=i}^g r_l. \quad (9)$$

The right hand side is

$$\sum_{l=i}^g r_l = \sum_{l=i}^g -\Delta_{S_l} p(D_G^{(l-1)}) = \sum_{l=i}^g (p(D_G^{(l-1)}) - p(D_G^{(l)})) = p(D_G^{(i-1)}) - p(D_G^{(g)}) = p(D_G^{(i-1)}) - 1.$$

So, proving (9) is equivalent to proving

$$\sum_{j=1}^h p'_{D_G^{(i-1)}}(Y_j) \geq p(D_G^{(i-1)}) - 1. \quad (10)$$

This inequality can be illustrated by Fig.3. In this figure, OPT is decomposed into four stars. A comprehension for the value $p'_D(Y_1) = 3$ is that in Fig.3(c), the double circled components are merged into the triangled component. Call the new component as \tilde{C} . Then, the comprehension of $p'_D(Y_2) = 3$ is that in Fig.3(d), double circled components are merged into the triangled component. Notice that this triangled component is contained in \tilde{C} , and thus we can regard it as \tilde{C} in our comprehension. Continue this procedure sequentially in such a way that $Y_1 \cup \dots \cup Y_l$ is connected for $l = 1, \dots, 4$. Finally, all components of $G[D]$ are merged into one component, the reduction on the number of components is $p(D) - 1$. Notice that the inequality might be strict because some components are counted more than once in the summation part. For example, the component labeled by u_4 is repetitively counted.

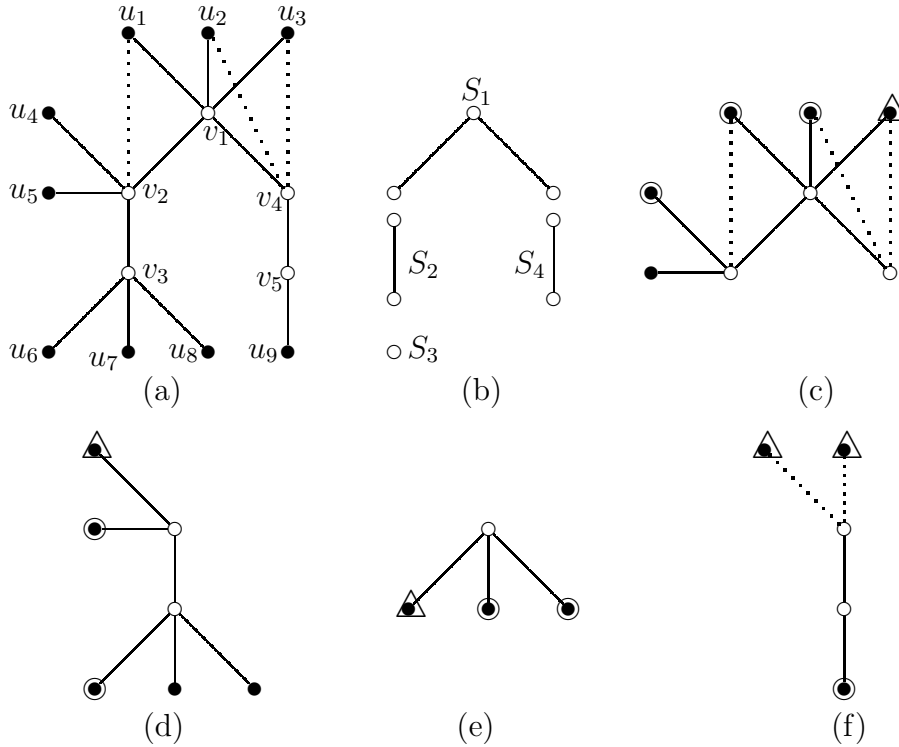


Figure 3: An illustration for inequality (10). Solid circles indicate components.

Claim 2 is proved.

By Lemma 5.7 (iv), we have

$$\sum_{i=1}^h c(Y_i) \leq 2opt. \quad (11)$$

Then by Claim 1, Claim 2, and the observation that $a_{0,j} \leq \delta$, we have

$$c(D_2) \leq \sum_{j=1}^h c(Y_j)H(\delta - 1) \leq 2H(\delta - 1)_{opt}.$$

The theorem is proved. \square

Combining Theorem 5.3 and Theorem 5.8, we have the following result.

Theorem 5.9. *Algorithm 1 is a polynomial-time $(H(\delta + m) + 2H(\delta - 1))$ -approximation for the m -MWCDs problem.*

6 Implementation on Unit Disk Graphs

Notice that the maximum degree δ in the performance ratio of the second part of the algorithm comes from $a_{0,j} \leq \delta$. So, to improve the approximation factor when the graph under consideration is a unit disk graph, we improve the upper bound for $a_{0,j}$ first. We use notation T_1, \dots, T_l in the proof of Lemma 5.7. Recall that each T_i has $V(T_i) \subseteq C^* \setminus C$ and is decomposed into some stars, and the final decomposition of $C^* \setminus C$ is the union of these stars and a set of pruned nodes. We shall use $\|\cdot\|$ to denote Euclidean length.

Lemma 6.1. *In a unit disk graph, there exists a set of subtrees T_1, \dots, T_l in the proof of Lemma 5.7 such that any node $v \in V(T_i)$ has $|NC_C(v)| + \deg_{T_i}(v) \leq 5$.*

Proof. Construct a spanning tree of $G[C \cup C^*]$ as follows. First, replace each component of $G[C]$ by a spanning tree of that component, which is called a *tree component*. Then, find a minimum length tree T which spans all nodes of $C^* \setminus C$ and all tree components, where “minimum length” is with respect to Euclidean distance. For convenience of statement, we shall call a tree which spans all nodes of $C^* \setminus C$ and all tree components as a *valid tree*. Each component of $G[C]$ is called a component node of such a tree and is dealt with as a whole in the following. Similarly to the proof of Lemma 5.7, by pruning leafs in $C^* \setminus C$ and by splitting off at component nodes, we obtain a set of full components T'_1, \dots, T'_l . Let T_i be the subtree of T'_i with component nodes removed. We choose tree T such that

$$\sum_{i=1}^l \sum_{u \in V(T_i)} \deg_{T_i}(u) \text{ is as small as possible.} \quad (12)$$

Suppose there is a node $v \in V(T_i)$ with $|NC_C(v)| + \deg_{T_i}(v) = t > 6$, assume $NC_C(v) \cup N_{T_i}(v) = \{x_1, \dots, x_t\}$, where x_j is a node in $C^* \setminus C$ or a node in a component neighbor of v (if v is adjacent with more than one node of a component neighbor, only one node of that component neighbor is chosen to appear in $\{x_1, \dots, x_t\}$), and x_1, \dots, x_t are in a clockwise order around node v . Since $t > 6$, there is an index j with $\angle x_j v x_{j+1} < \pi/3$ (x_{t+1} is viewed as x_1). Then $\|x_j x_{j+1}\| < \max\{\|vx_j\|, \|vx_{j+1}\|\}$, say $\|x_j x_{j+1}\| < \|vx_j\|$.

Replacing edge vx_j by x_jx_{j+1} , we obtain another valid tree whose Euclidean length is shorter than T , a contradiction. So,

$$\text{every node } v \in V(T_i) \text{ has } |NC_C(v)| + \deg_{T_i}(v) \leq 6. \quad (13)$$

A node u with $|NC_C(u)| + \deg_{T_i}(u) = 6$ is called *bad*.

Similar argument shows that for any bad node $v \in V(T_i)$, $\angle x_jvx_{j+1} = \pi/3$ for $j = 1, \dots, 6$ and $\|vx_1\| = \dots = \|vx_6\|$. In other words, x_1, \dots, x_6 locate at the corners of a regular hexagon with center v . First, suppose node v has a component neighbor, say x_1 is in a component neighbor of v . If $x_2 \in C$, then x_2 must be in a same component of $G[C]$ as x_1 , because $\|x_1x_2\| = \|vx_1\| \leq 1$, contradicting our convention that one component has at most one node appearing in $\{x_1, \dots, x_6\}$. So, $x_2 \in C^* \setminus C$. Then, $\tilde{T} = T - \{vx_2\} + \{x_1x_2\}$ is a valid tree with the same length as T . Notice that $\sum_{i=1}^l \sum_{u \in V(T_i)} \deg_{T_i}(u)$ is decreased by two (edge x_1x_2 does not contribute to the degree sum since one of its end is in a component neighbor and thus does not belong to T_i), contradicting the choice of T (see (12)). So, for $j = 1, \dots, 6$, $x_j \in C^* \setminus C$. Since $\tilde{T} = T - \{vx_{j-1}, vx_{j+1}\} + \{x_jx_{j-1}, x_jx_{j+1}\}$ is also a valid tree whose length is the same as T , we have $|NC_C(x_j)| + \deg_{\tilde{T}_i}(x_j) \leq 6$. By noticing that $\deg_{\tilde{T}_i}(x_j) = \deg_{T_i}(x_j) + 2$ (since both $x_{j-1}, x_{j+1} \in C^* \setminus C$), we have

$$|NC_C(x_j)| + \deg_{T_i}(x_j) \leq 4. \quad (14)$$

This inequality holds for any $j = 1, \dots, 6$. Notice that $\hat{T} = T - \{vx_1\} + \{x_1x_2\}$ is a valid tree with the same length as T and $\sum_{i=1}^l \sum_{u \in V(\hat{T}_i)} \deg_{\hat{T}_i}(u) = \sum_{i=1}^l \sum_{u \in V(T_i)} \deg_{T_i}(u)$. By property (13) and (14), we have

$$|NC_C(u)| + \deg_{\hat{T}_i}(u) = \begin{cases} |NC_C(u)| + \deg_{T_i}(u) - 1 \leq 5, & u = v, \\ |NC_C(u)| + \deg_{T_i}(u) + 1 \leq 5, & u = x_1 \text{ or } x_2, \\ |NC_C(u)| + \deg_{T_i}(u) \leq 6, & u \neq v, x_1, x_2. \end{cases}$$

So, the number of bad nodes in \hat{T} is strictly reduced. By recursively executing such an operation, we have a tree satisfying the requirement of this lemma.

Recall that condition (6) plays an important role in the decomposition. This does not pose any difficulty here, because by the modification method in the proof of Lemma 5.7, if $u \in C^* \setminus C$ is not adjacent with any component neighbor in T' , then we may just add an edge between u and a component neighbor, and remove an edge between u and another node in $C^* \setminus C$. Such an operation does not increase the value of $|NC_C(u)| + \deg_{T_i}(u)$. \square

Theorem 6.2. *When applied to unit disk graphs, the node set D_2 produced by Algorithm 1 has cost $c(D_2) \leq 3.67\text{opt}$.*

Proof. Let T_1, \dots, T_l be the subtrees in Lemma 6.1. Choose node $v_i = \arg \max\{c(v) : v \in V(T_i)\}$ to be the root of T_i . Decompose $C^* \setminus C$ as in the proof of Lemma 5.7. Let $\{Y_v^{(i)}\}$ be the set of stars coming from the decomposition of T_i . To avoid ambiguity, we use $a_{0, Y_v^{(i)}}$ to denote $p'_{D_1}(Y_v^{(i)})$ (which is $a_{0,j}$ in the proof of Theorem 5.8). By Lemma 6.1, if $v = v_i$, then

$a_{0,Y_v^{(i)}} \leq |NC_{D_1}(v)| + \deg_{T_i}(v) \leq 5$; if $v \neq v_i$, then $a_{0,Y_v^{(i)}} \leq |NC_{D_1}(v)| + \deg_{T_i}(v) - 1 \leq 4$ (this is because the parent of v is not in Y_v if $v \neq v_i$).

Notice that v_i belongs to exactly one star in the decomposition of $C^* \setminus C$. Hence inequality (11) can be improved to

$$\sum_{i=1}^l \sum_{Y_v^{(i)}} c(Y_v^{(i)}) + \sum_{i=1}^l c(v_i) \leq 2opt. \quad (15)$$

Combining Lemma 6.1 with Claim 1 and Claim 2 of Theorem 5.8,

$$c(D_2) \leq \sum_{i=1}^l \sum_{Y_v^{(i)}} c(Y_v^{(i)}) H(a_{0,Y_v^{(i)}} - 1) \leq \sum_{i=1}^l \left(H(4)c(Y_{v_i}^{(i)}) + H(3) \sum_{Y_v^{(i)}, v \neq v_i} c(Y_v^{(i)}) \right). \quad (16)$$

Since D_1 is an m -fold dominating set, every v_i has at least one component neighbor. Then by Lemma 6.1, $\deg_{T_i}(v_i) \leq 4$, and thus $Y_{v_i}^{(i)}$ has at most five nodes. Since v_i has the maximum cost in T_i , we have $c(Y_{v_i}^{(i)}) \leq 5c(v_i)$. So,

$$H(4)c(Y_{v_i}^{(i)}) = H(3)c(Y_{v_i}^{(i)}) + \frac{c(Y_{v_i}^{(i)})}{4} \leq H(3)c(Y_{v_i}^{(i)}) + 5c(v_i)/4 < H(3)(c(Y_{v_i}^{(i)}) + c(v_i)).$$

Combining this inequality with (15) and (16),

$$c(D_2) \leq H(3) \left(\sum_{i=1}^l \sum_{Y_v^{(i)}} c(Y_v^{(i)}) + \sum_{i=1}^l c(v_i) \right) \leq 2H(3)opt < 3.67opt.$$

The theorem is proved. \square

We have presented a $(3+\varepsilon)$ -approximation algorithm for the Minimum Weight m -Fold Dominating Set problem in unit disk graph. Based on this result, we have the following performance ration.

Theorem 6.3. *Replacing the first part of Algorithm 1 by the $(3+\varepsilon)$ -approximation algorithm in [26] yields a $(6.67+\varepsilon)$ -approximation for the m -MWCDS problem.*

7 Conclusion

In this paper, we presented a $(H(\delta+m) + 2H(\delta-1))$ -approximation algorithm for the minimum weight $(1, m)$ -CDS problem, where δ is the maximum degree of the graph. Compared with the $1.35 \ln n$ -approximation algorithm for the minimum $(1, 1)$ -CDS problem [11], our constant is larger. However, since in many cases, the maximum degree is much smaller than the number of nodes, our result is an improvement on the performance ratio in some sense. In particular, the replacement of n by δ in the performance ratio makes

it possible to obtain a 3.67-approximation for the connecting part when the topology of the network is a unit disk graph. In fact, Zou *et al.* obtained a 2.5ρ -approximation for the connecting part in a unit disk graph, where ρ is the performance ratio for the minimum Steiner tree problem. If the best $\rho = 1.39$ is used, their algorithm has performance ratio 3.475. Notice that the 1.39-approximation algorithm for the minimum Steiner tree problem uses a combination of iterated rounding and random rounding. Our algorithm has the advantage of purely combinatorial and deterministic. Furthermore, we expect our method to have a theoretical value which can be used to deal with other problems.

Acknowledgements

This research is supported by NSFC (61222201, 11531011), SRFDP (20126501110001), and Xingjiang Talent Youth Project (2013711011). It is accomplished when the second author is visiting National Chiao Tung University, Taiwan, sponsored by “Aiming for the Top University Program” of the National Chiao Tung University and Ministry of Education, Taiwan.

References

- [1] Byrka J, Grandoni F, Rothvoss T, Sanità L (2013) J. ACM, 60(1), Article 6.
- [2] Cheng X, Huang X, Li D, Wu W, Du DZ (2003) A polynomial-time approximation scheme for the minimum-connected dominating set in ad hoc wireless networks. *Networks*, 42(4), 202–208.
- [3] Dai F, Wu J (2006) On constructing k -connected k -dominating set in wireless ad hoc and sensor networks. *J Parallel Distrib Comput* 66(7), 947–958.
- [4] Das B, Bharghavan V (1997) Routing in ad-hoc networks using minimum connected dominating sets. *IEEE International Conference on Communications*, Montreal, 376–380.
- [5] Du DZ, Graham RL, Pardalos PM, Wan PJ, Wu WL, Zhao W (2008) Analysis of greedy approximation with nonsubmodular potential functions. *SODA’08*, 167–175.
- [6] Du DZ, Ko K, Hu X (2012) Design and analysis of approximation algorithms. Springer, New York.
- [7] Du DZ, Wan PJ (2013) Connected Dominating Set: Theory and Applications, Springer, New York.
- [8] Ephremides A, Wieselthier J, Baker D (1987) A design concept for reliable mobile radio networks with frequency hopping signaling. *Proc IEEE* 56–73.

- [9] Gao X, Wang Y, Li X, Wu W (2009) Analysis on theoretical bounds for approximating dominating set problems. *Discrete Math. Algo. Appl.*, 1(1), 71–84.
- [10] Guha S, Khuller S (1998) Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4), 374–387.
- [11] Guha S, Khuller S (1999) Improved methods for approximating node weighted steiner trees and connected dominating sets. *Information and Computation* 150, 57–74.
- [12] Li M, Wan P, Yao F (2009) Tighter approximation bounds for minimum CDS in wireless ad hoc networks. *ISAAC’09, LNCS 5878*, 699–709.
- [13] Li Y, Wu Y, Ai C, Beyah R (2012) On the construction of k -connected m -dominating sets in wireless networks. *J. Comb. Optim.* 23, 118–139.
- [14] Robins G, Zelikovsky A (2000) Improved Steiner tree approximation in graphs. *SODA’00*, 770–779.
- [15] Ruan L, Du H, Jia X, Wu W, Li Y, Ko K (2004) A greedy approximation for minimum connected dominating sets. *Theor. Comput. Sci.* 329(1), 325–330.
- [16] Shang W, Yao F, Wan P, Hu X (2008) On minimum m -connected k -dominating set problem in unit disc graphs. *J. Comb. Optim.* 16, 99–106.
- [17] Shi Y, Zhang Y, Zhang Z, Wu W (2014) A greedy algorithm for the minimum 2-connected m -fold dominating set problem. *J. Comb. Optim.* doi:10.1007/s10878-014-9720-6
- [18] Thai M, Zhang N, Tiwari R, Xu X (2007) On approximation algorithms of k -connected m -dominating sets in disk graphs. *Theor. Comput. Sci.* 385, 49–59.
- [19] Wan PG, Alzoubi KM, Frieder O (2002) Distributed construction of connected dominating set in wireless ad hoc networks. *INFOCOM’02* 1597–1604, *Mobile Networks and Applications*, 9, 141–149.
- [20] Wan PJ, Wang L, Yao F (2008) Two-phased approximation algorithms for minimum CDS in wireless ad hoc networks. *ICDCS’08*, 337–344.
- [21] Wang F, Thai M, Du DZ (2009) On the construction of 2-connected virtual backbone in wireless networks. *IEEE Trans. Wirel. Commun.* 8(3), 1230–1237.
- [22] Wang W, Kim D, An M, Gao W, Li X, Zhang Z, Wu W (2013) On construction of quality fault-tolerant virtual backbone in wireless networks. *IEEE/ACM Trans. Netw.* 21(5), 1499–1510.
- [23] Wang W, Liu B, Kim D, Li D, Wang J, Jiang Y (2015) A better constant approximation for minimum 3-connected m -dominating set problem in unit disk graph using Tutte decomposition, to appear in *INFOCOM’15*.

- [24] Wu W, Du H, Jia X, Li Y, Huang S (2006) Minimum connected dominating sets and maximal independent sets in unit disk graphs. *Theor. Comput. Sci.* 352, 1–7.
- [25] Zhang Z, Gao X, Wu W, Du DZ (2009) A PTAS for minimum connected dominating set in 3-dimensional wireless sensor networks. *J. Glob. Optim.* 45, 451–458.
- [26] Zhang Z, Willson J, Wu W, Zhu X, Du DZ (2015) $(3 + \varepsilon)$ -approximation for the minimum weight k -cover problem, in submission. A preliminary version was *Computer Communications (INFOCOM)*, 2015 IEEE Conference on, pp 1364–1372.
- [27] Zhang Z, Liu Q, Li D (2009) Two algorithms for connected r -hop k -domination set. *Discrete Math. Algo. Appl.* 1(4), 485–498.
- [28] Zhang Z, Shi Y (2015) Approximation algorithm for minimum weight fault-tolerant virtual backbone in homogeneous wireless sensor network. *Computer Communications (INFOCOM)*, 2015 IEEE Conference on, pp 1080–1085.
- [29] Zhou J, Zhang Z, Wu W, Xing K (2014) A greedy algorithm for the fault-tolerant connected dominating set in a general graph. *J. Comb. Optim.* 28(1), 310–319.
- [30] Zou F, Li X, Gao S, Wu W (2009) Node-weighted Steiner tree approximation in unit disk graphs. *J. Comb. Optim.* 18, 342–349.
- [31] Zou F, Wang Y, Xu X, Du H, Li X, Wan P, Wu W (2011) New approximations for weighted dominating sets and connected dominating sets in unit disk graphs. *Theoret. Comput. Sci.* 412(3), 198–208.